CAPÍTULO 2
AMBIENTE DE PROGRAMAÇÃO

2.1. INSTALAÇÃO DO JDK

O JDK (Java Development Kit) é o kit de desenvolvimento Java, aquele utilizado para criar os programas e não somente executá-los.

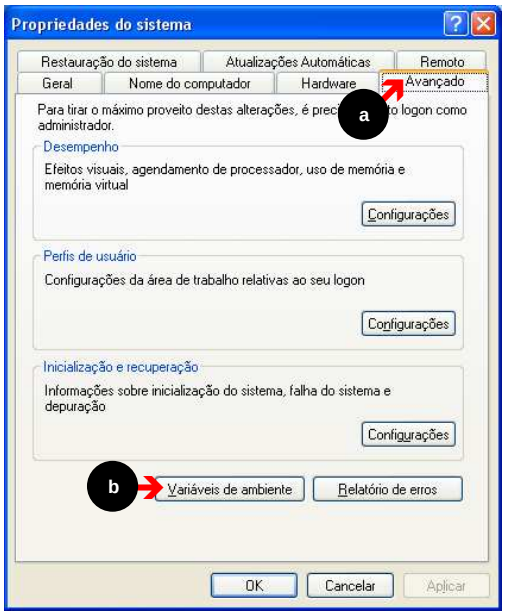
1 Baixe o arquivo **Java Plataform (JDK)** no site:

http://www.oracle.com/technetwork/java/javase/downloads/index.html

Observação: Acesse a pasta *Softwares* no moodle e copie o arquivo *Java JDK 6*.

2 Utilizando o Windows Explorer, vá até a pasta onde estiver o instalador Java e efetue a instalação do JDK.

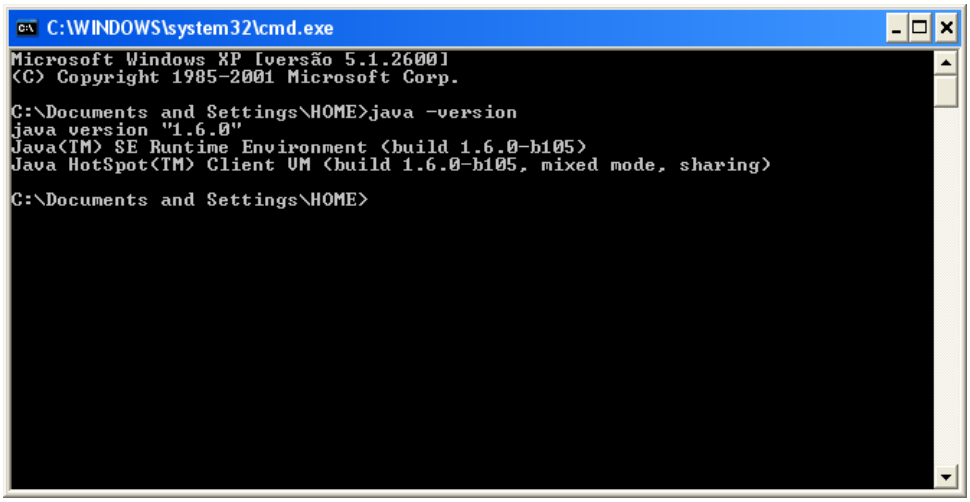
3 A configuração do ambiente é onde os desenvolvedores iniciantes tem a maior dificuldade, que é o processo de configuração das Variáveis de Ambiente:
3.1. Clique com o botão direito do mouse no ícone Meu Computador;
3.2. Escolha a opção Propriedades;
3.3. Clique na guia Avançado e, depois, no botão Variáveis de Ambiente;



3.4. Na opção **Variáveis do Sistema**, efetue um duplo-clique na variável *Path*. Acrescente ao final da variável um ponto-e-vírgula e o caminho onde o JDK foi instalado. **Exemplo:** ;C:\Arquivos de programas\Java\jdk1.6.0\bin

4 Para verificar se a instalação e configuração do JDK foi efetuada corretamente, abra o prompt de comando do Windows e digite:

java -version



2.2. INTEGRATED DEVELOPMENT ENVIRONMENT

IDE (Integrated Development Environment, ou ambiente integrado de desenvolvimento) é um programa de computador que reúne recursos e ferramentas de apoio para agilizar o desenvolvimento de softwares. Entre estas ferramentas geralmente temos:

- Um editor de texto com facilidades especialmente desenhadas para a linguagem;
- Um compilador e um interpretador, no caso de Java e outras linguagens interpretadas;
- Um editor gráfico, com facilidades para criação e edição da interface gráfica do programa a ser desenvolvido;

- Um debugger, uma ferramenta especialmente feita para se tirar os bugs do código. Ela possibilita um monitoramento mais elegante do funcionamento do seu programa, facilitando a detecção e remoção dos erros.

2.1.1. Eclipse (www.eclipse.org)

O Eclipse é uma robusta plataforma de software livre para desenvolvimento de aplicações em Java. O projeto Eclipse surgiu e mantém sua força da doação e envolvimento da IBM para manter como código aberto uma base para suas ferramentas IBM/Rational. O projeto, ao longo dos anos, ganhou o apoio e participação de grandes empresas e instituições, como Borland, BEA, JBoss e Oracle. É mantido pela Eclipse Foundation, que possui projetos que se concentram na criação de uma plataforma de desenvolvimento composta por quadros extensíveis e ferramentas para construção, implantação e gerenciamento de softwares em todo o seu ciclo de vida.

2.1.2. NetBeans (www.netbeans.org)

Ambiente de desenvolvimento java cujo projeto é fortemente patrocinado pela Sun. Costuma ser a primeira IDE a incorporar novos recursos de novas versões do java e disputa com o Eclipse o primeiro lugar entre as ferramentas livres para desenvolvimento comercial. Possui um ótimo editor visual de interface gráfica Swing e ótimas facilidades para desenvolvimento em Java. Diferente do Eclipse, o resultado objetivado pelo projeto NetBeans sempre foi um ambiente de desenvolvimento integrado (IDE) Java completo e bem acabado.

2.1.3. JDeveloper (oracle.com/technology/products/jdev/)

JDeveloper é um ambiente de desenvolvimento integrado gratuito (desde 2005) da Oracle que oferece funcionalidades para o desenvolvimento em Java, XML, SQL e PL/SQL, HTML, JavaScript, BPEL e PHP. O JDeveloper cobre todo o ciclo de desenvolvimento desde a análise até a codificação, a manutenção, a otimização e a implantação.

ORACLE
JDEVELOPER

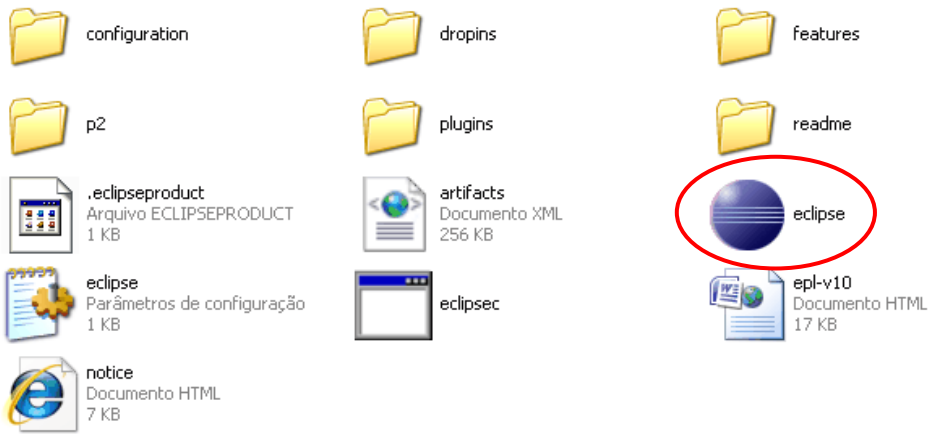
2.3. IDE ECLIPSE

O Ambiente de desenvolvimento que iremos trabalhar é o Eclipse que possui facilidades que vão desde a rápida visualização de todos os arquivos contidos no projeto até ferramentas de gerenciamento de trabalho coletivo. Você perceberá que ele apenas gera trechos de códigos óbvios, sempre ao seu comando. Existem também centenas de plugins gratuitos para gerar diagramas UML, suporte a servidores de aplicação, visualizadores de banco de dados e muitos outros. A versão do Eclipse escolhida é a Eclipse Helios encontrada no site *www.eclipse.org* na seção ***Downloads***, ***Older Versions*** ou na unidade C: dos laboratórios de informática.

2.1.4. Instalação

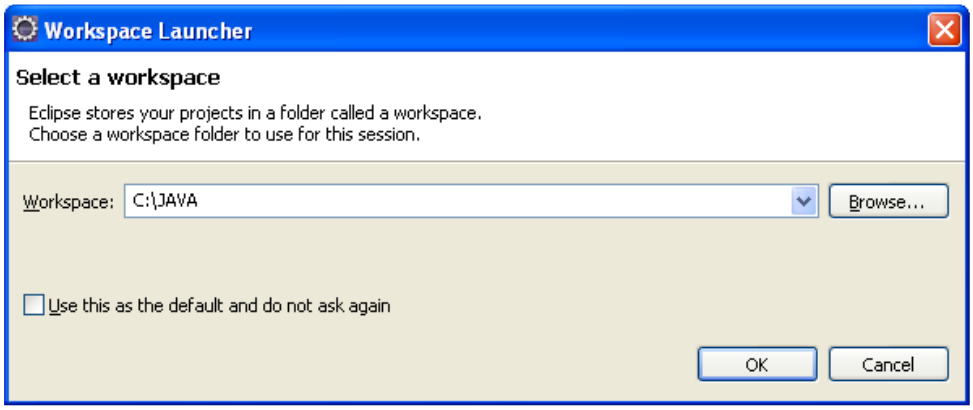
1. O Eclipse Helios não possui instalador. Após baixar o arquivo, descompacte-o e copie a pasta para o diretório C:\Arquivos de Programas.

2. Crie um atalho do arquivo Eclipse na Área de Trabalho.



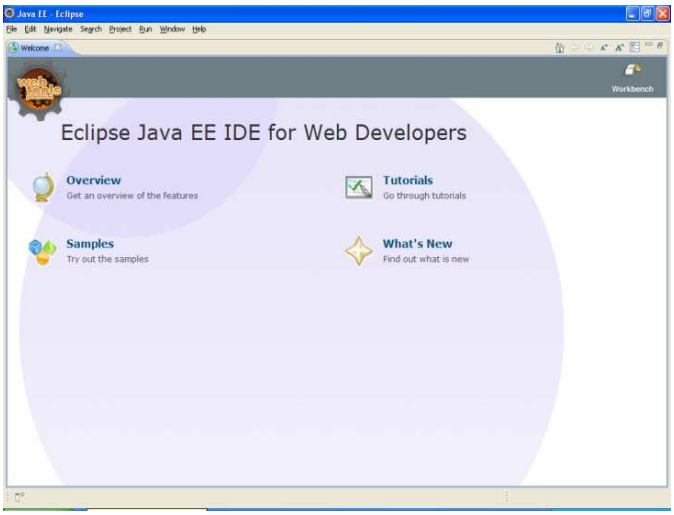
2.1.5. Workspace

Workspace define o diretório em que contém as configurações pessoais e os projetos serão gravados. Logo que é executado, o Eclipse permite que o usuário selecione uma pasta como Workspace.



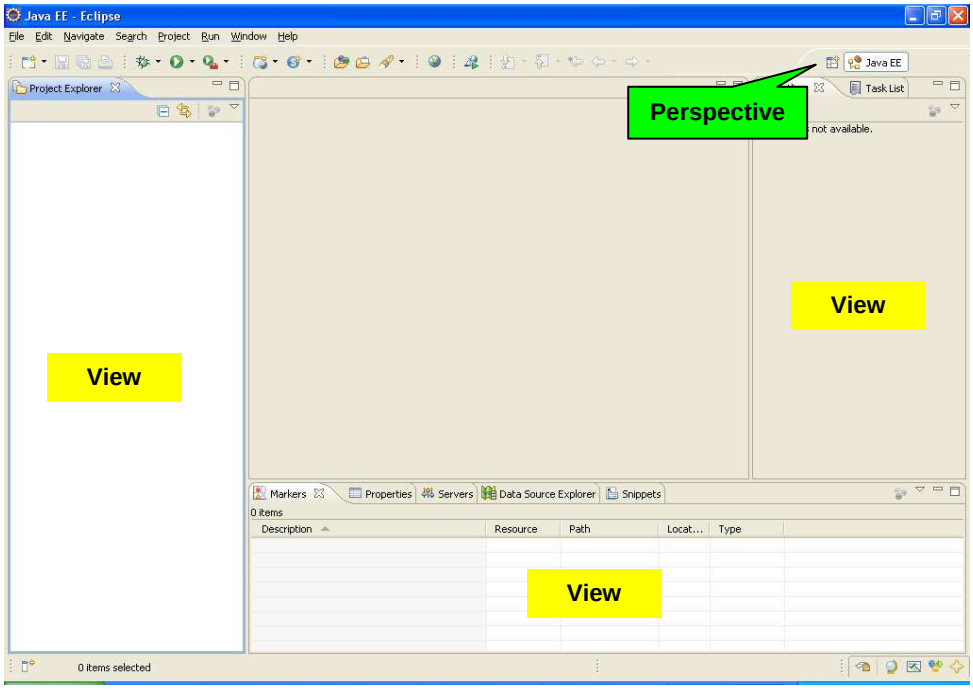
2.1.6. Welcome

A primeira tela do Eclipse (welcome) mostra "links" para alguns exemplos, tutorias, visão geral da ferramenta e novidades.



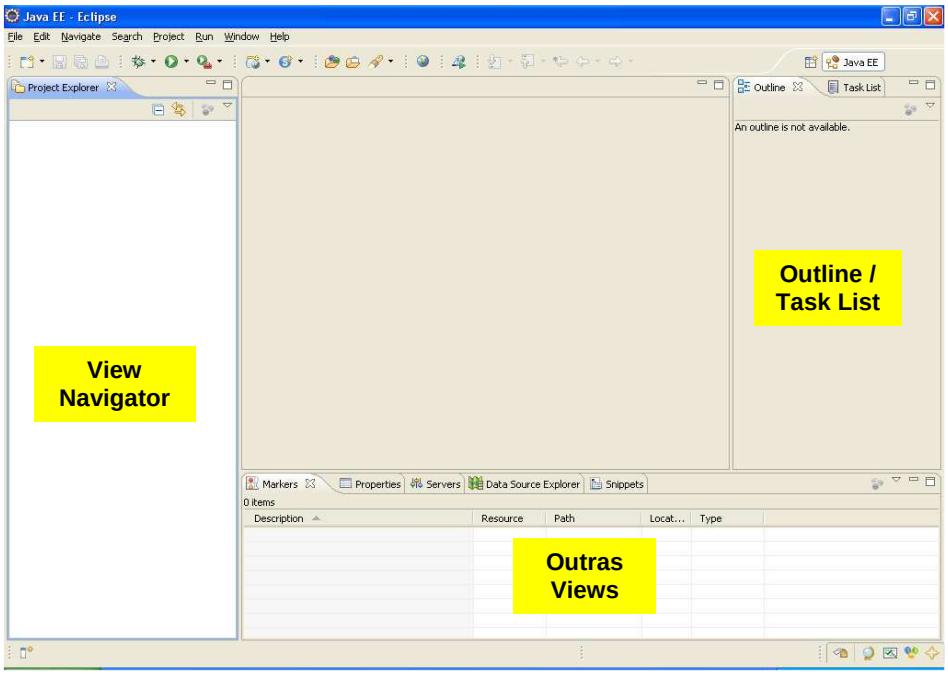
2.1.7. Workbench

Da tela Welcome podemos ir para Workbench que é a área de trabalho do Eclipse clicando no ícone deste ou simplesmente fechando a tela Welcome.



2.1.8. Views

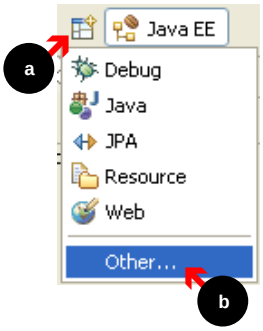
As telas que são mostradas na workbench são chamadas de views. O desenvolvedor pode abrir, fechar ou mover qualquer view ao seu gosto e necessidade. Uma nova view pode ser aberta através do menu **WINDOW ➔ SHOW VIEW ➔ OTHER**.

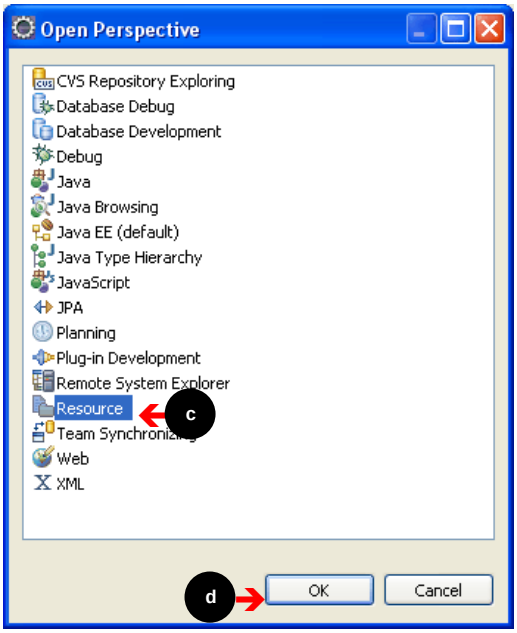


2.1.9. Perspective

O Eclipse oferece vários modos de trabalho ao desenvolvedor. Cada modo de trabalho é adequado para algum tipo de tarefa. Esses modos de trabalhos são chamados de perspectives (perspectivas). Podemos abrir uma perspectiva através do ícone no canto superior direito da Workbench.

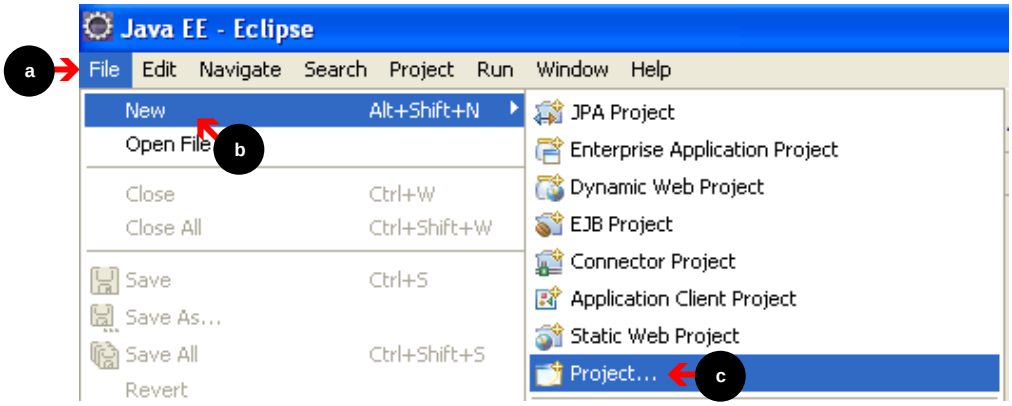
Neste momento, trabalharemos com a perspectiva Resource, antes da de Java, pois ela possui um conjunto de Views mais simples. Mude para a perspectiva Resource, clicando no ícone ao lado da perspectiva Java, selecionando Other e depois Resource.



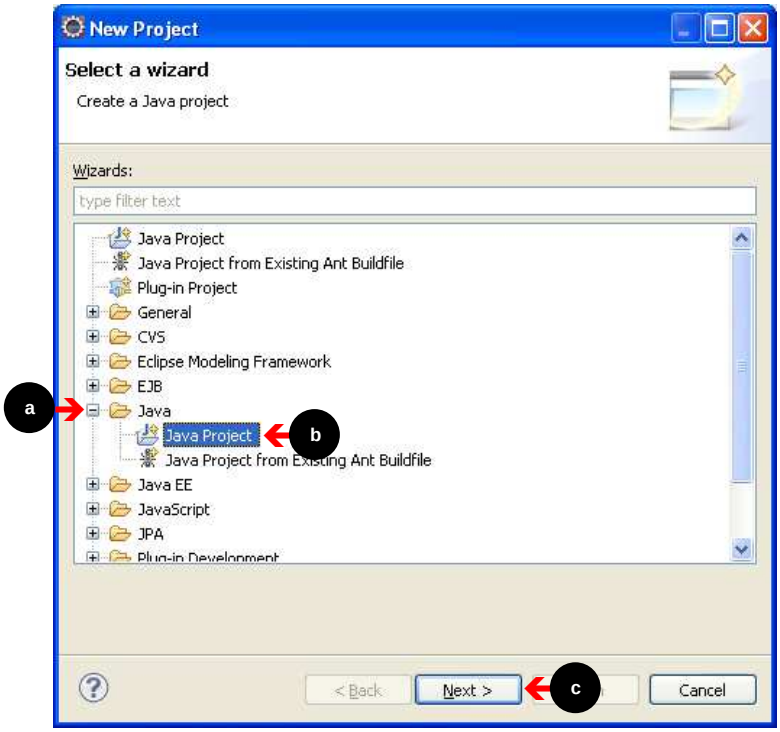


2.1.10. Criação de Projeto

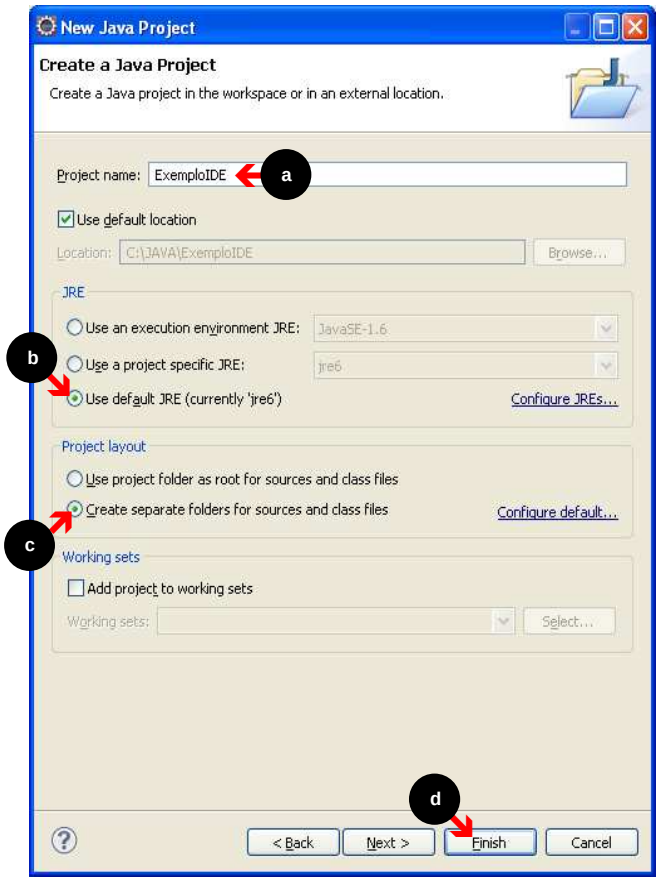
1 Acesse o menu **FILE** ➔ **NEW** ➔ **PROJECT**...



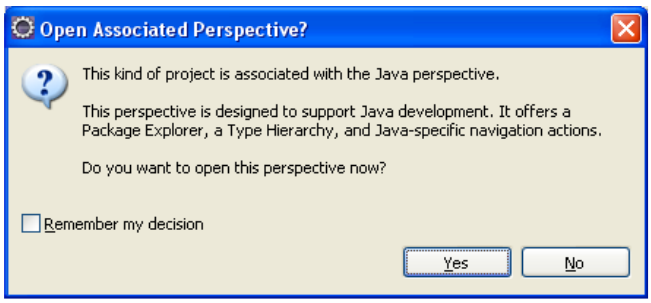
2 Selecione a opção Java Project na pasta Java. Clique no botão **NEXT**.



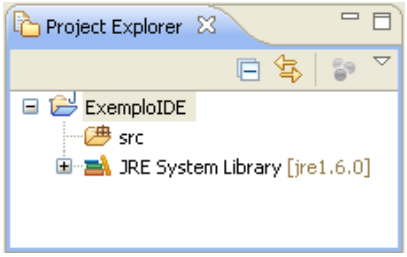
3 Na tela de criação de projeto java devemos escolher um nome para o projeto. Marque a opção "*USE DEFAULT JRE*" e, depois, marque também a opção "*CREATE SEPARATE SOURCE AND OUTPUT FOLDERS*", desta maneira seus arquivos .java e arquivos .class estarão em diretórios diferentes, para você trabalhar de uma maneira mais organizada. Finalize a criação.



4 O Eclipse pedirá para trocar a perspectiva para Java. Escolha "No" para permanecer em Resource.

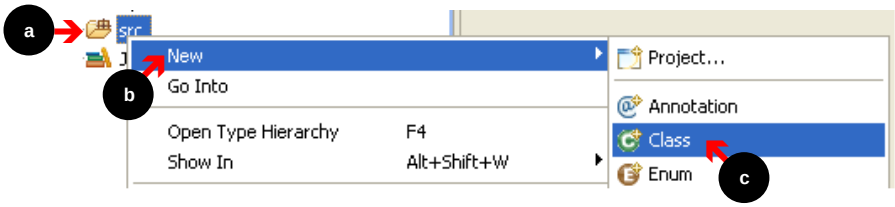


5 Agora, na **VIEW NAVIGATOR**, veremos o novo projeto e suas pastas e arquivos.

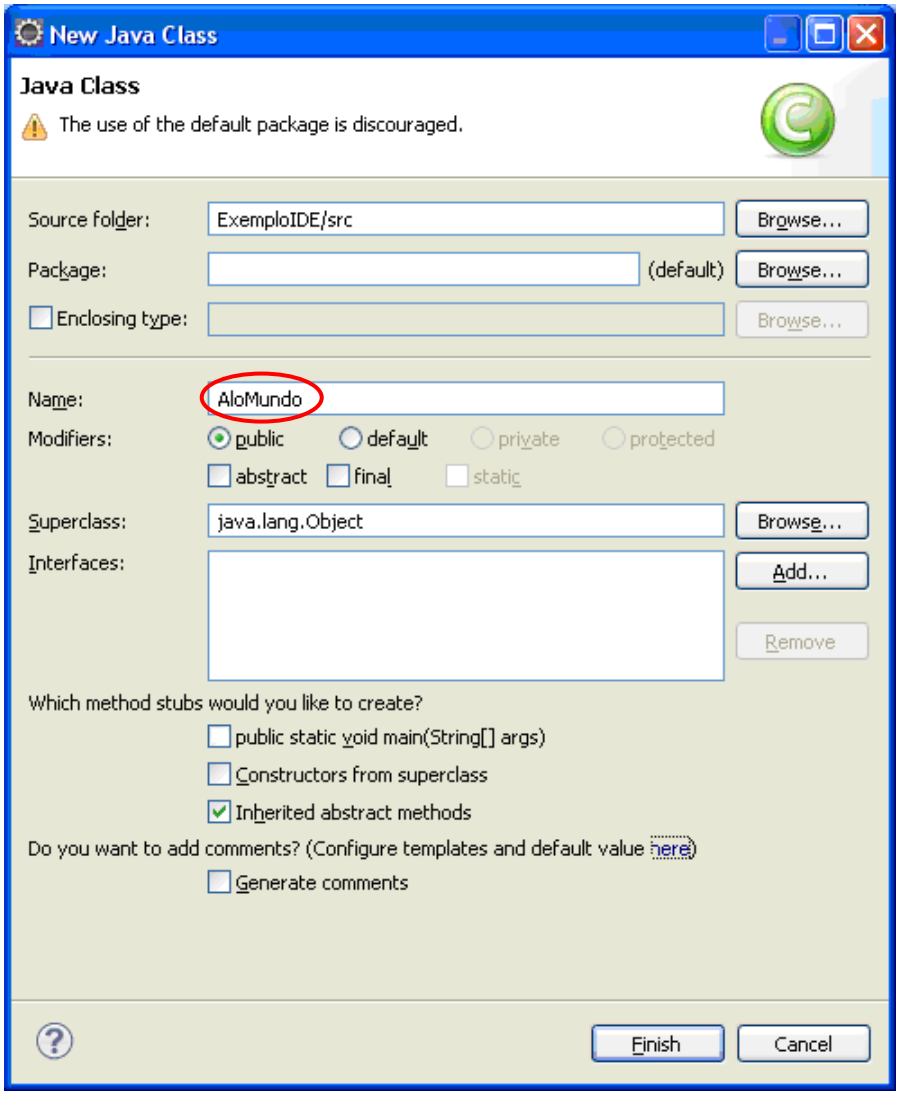


2.1.11. Criação de Classes

1 Após a criação de um projeto podemos criar uma classe. Clique com o botão direito na pasta SRC, escolha a opção **NEW** ➔ **CLASS**.



2 Na tela de criação de classe devemos escolher um nome e clicar em **FINISH**.



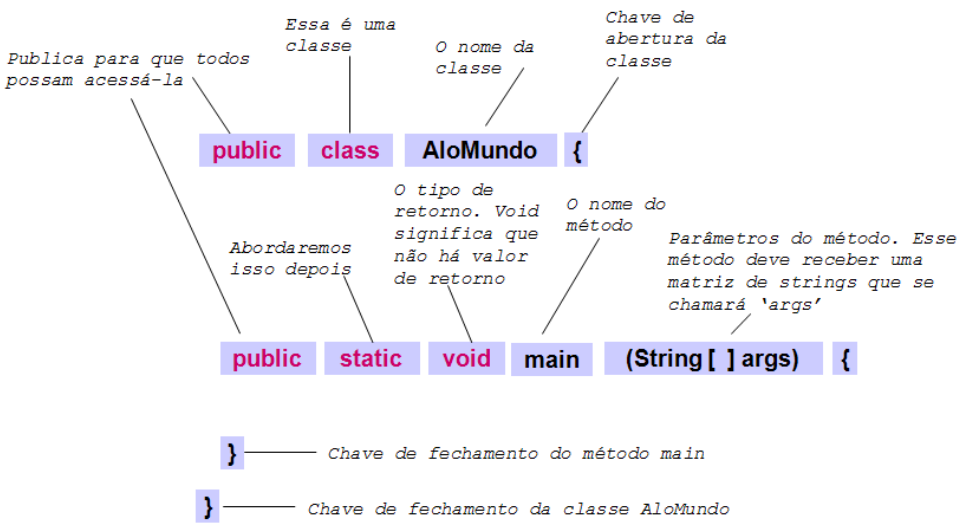
3 Um arquivo com o esqueleto da classe é criado na pasta src e automaticamente o Eclipse compila e salva o código compilado na pasta bin.

2.1.12. Criação da Classe Principal

Para um programa Java executar, é necessário definir um método especial para ser o ponto de entrada do programa, ou seja, para ser o primeiro método a ser chamado pela JVM quando o programa for executado. O método main se parece exatamente com:

```
public static void main(String[] args) {
    // SEU CÓDIGO FICA AQUI
}
```

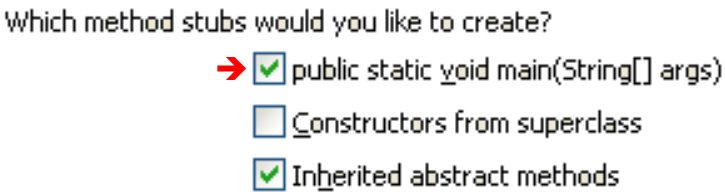
A JVM executará tudo que estiver entre as chaves { } do método principal. Todo aplicativo Java precisa ter pelo menos uma **classe** e um método **main** (não é um método main por classe, apenas um por aplicativo).



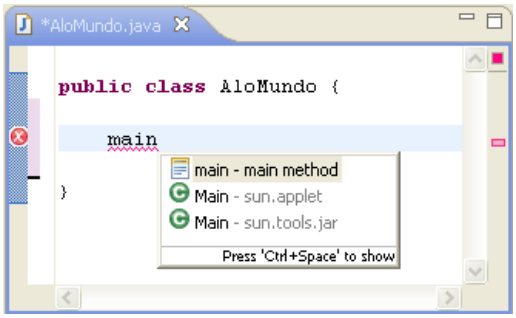
Em Java, tudo é inserido em uma classe. Criamos um arquivo de código fonte (com extensão .java) e, em seguida, o convertemosem um novo arquivo de classe (com extensão .class). Quando executamos um programa, na verdade estamos executando uma classe. Executar um programa significa informar à JVM para carregar a classe AloMundo e, em seguida, executar seu método main(). A JVM continuará executando até todo o código de main ter terminado. Resumindo, o método main() é onde o programa começará a ser executado.

Independentemente do tamanho de seu programa (em outras palavras, não importa quantas classes o seu programa vai usar), é preciso que haja um método main() que dê início ao processo.

O método main() poderá ser gerado na criação de uma classe ativando a opção "public static void main" após colocar o nome da classe.

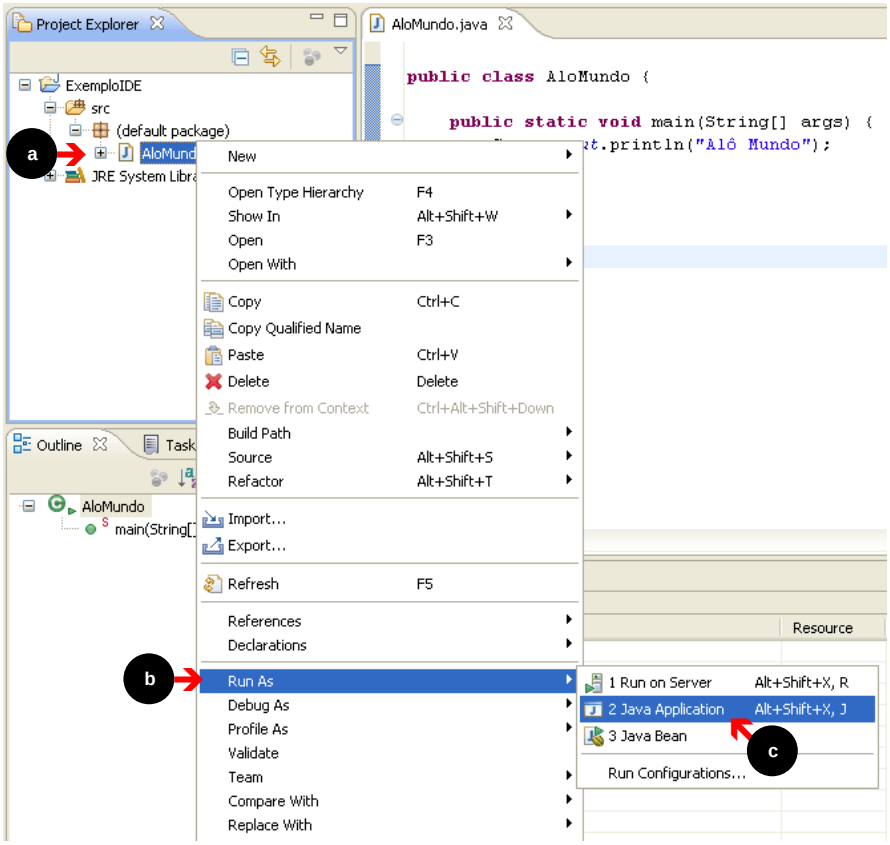


O método main() pode ser gerado utilizando Content Assist através do atalho CTRL+ESPACO. Basta digitar "main" seguido de CTRL+ESPACO e aceitar a sugestão do template para o método main.

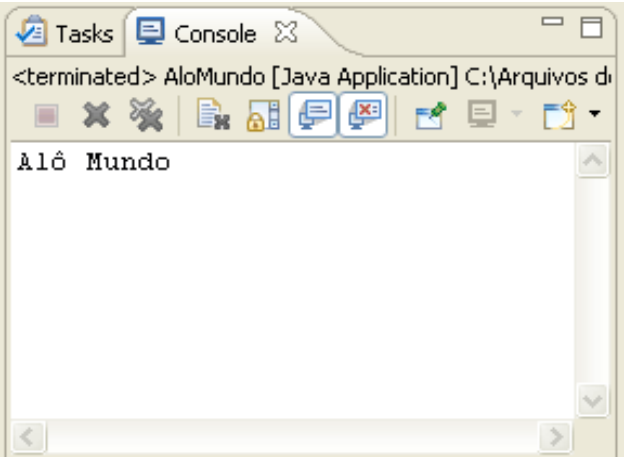


2.1.13. Executando o main

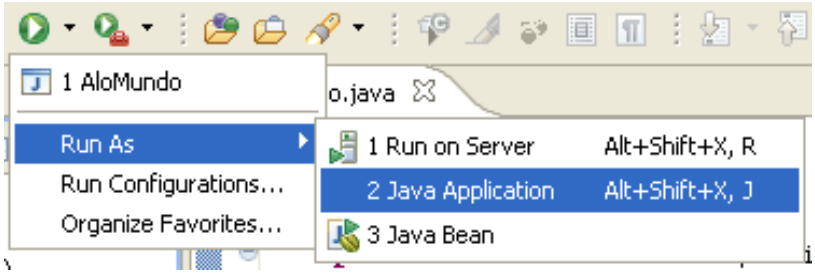
Podemos executar uma classe que possui o método main() através do botão direito do mouse na respectiva classe ➔ **RUN AS** ➔ **JAVA APPLICATION** ou pressionado CTRL + F11.



O Eclipse abrirá uma View chamada Console onde será apresentada a saída do seu programa:



Quando você precisar rodar de novo, basta clicar no ícone verde de play na toolbar, que roda o programa anterior. Ao lado desse ícone tem uma setinha onde são listados os 10 últimos executados.

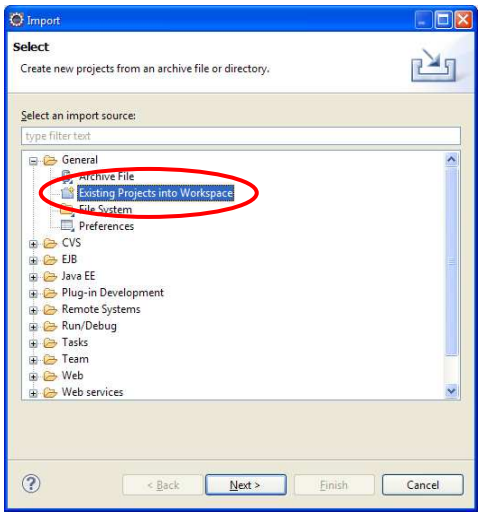


2.1.14. Abrindo projetos

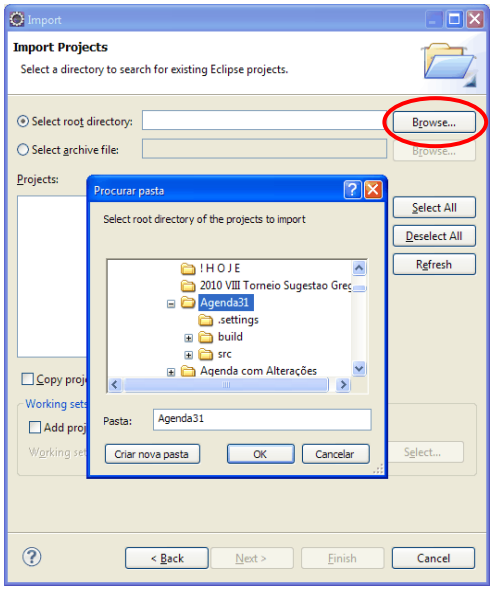
Para abrir um projeto criado no Eclipse é necessário importá-lo para a workspace. A seguir, temos as etapas necessárias para importar um projeto no Eclipse:

1. Acesse o menu **FILE** ➔ **IMPORT**

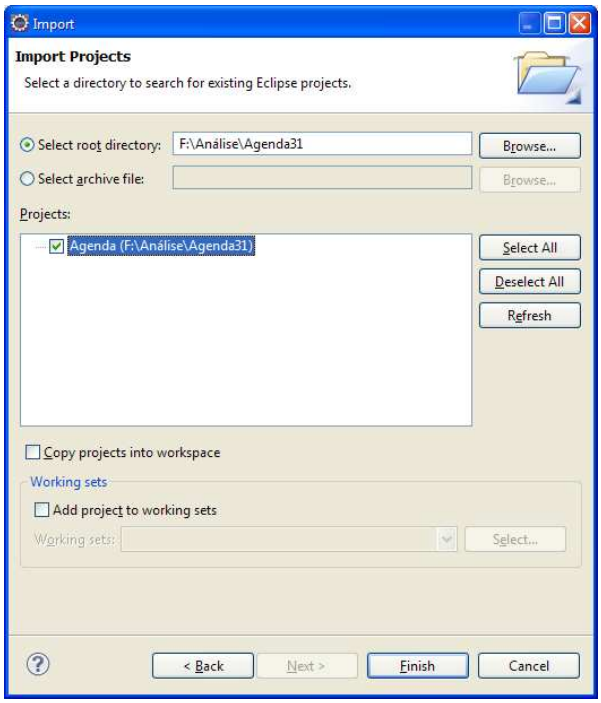
2. Localize a opção **GENERAL** ➔ **EXISTING PROJECTS INTO WORKSPACE**. Clique em **NEXT**.



3. Em "`Select root directory`", clique no botão **BROWSE**. Localize a pasta do projeto que deseja abrir e clique em **OK**.



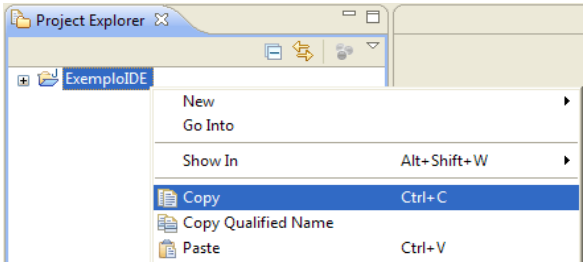
4 Selecione o projeto desejado (caso necessário) e clique no botão **FINISH**.



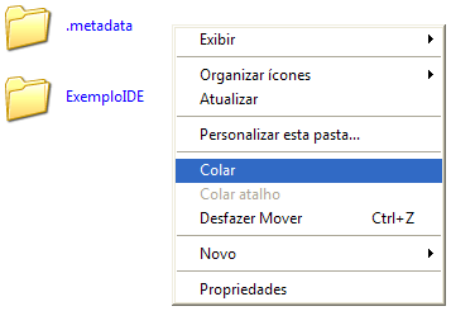
2.4. BACKUP DE PROJETOS

Para copiar os projetos para um local diferente da Workspace, basta copiálo e colá-lo no local desejado.

1 Clique com o botão direito do mouse no projeto e escolha **COPY**.



2 Acesse o local desejado para guardar o arquivo e cole-o.



2.5. ATALHOS

O Eclipse possui muitos atalhos úteis para o programador. Alguns bem interessantes de saber:

Ctrl + 1	Aciona o quick fixes com sugestões para correção de erros.
Ctrl + Espaço	Completa códigos.
Ctrl + 3	Aciona modo de descoberta de menu.
Ctrl + PgUp **Ctrl + PgDown**	Navega nas abas abertas. Útil quando estiver editando vários arquivos ao mesmo tempo.
Ctrl + Shift + F	Formata o código segundo as convenções do Java.
Ctrl + M	Expande a View atual para a tela toda (mesmo efeito de dar dois cliques no título da View).
Ctrl + Shift + L	Exibe todos os atalhos possíveis.
Ctrl + O	Exibe um outline para rápida navegação.
Ctrl + Shift + O	Organiza os import's.

2.6. EXERCÍCIOS: CRIANDO PROJETO E CLASSE

1. No Desktop, abra o Eclipse através do ícone existente. Quando perguntado sobre o diretório da Workspace, escolha a opção *C:/JAVA*.

2. Mude a Perspective para *Resource*.

3. Crie um novo projeto:
   a) Vá em **FILE** ➔ **NEW** ➔ **PROJECT**
   b) Escolha a opção **JAVA** ➔ **JAVA PROJECT**
   c) Digite *dsi-exercicio1* como nome do projeto, marque as opções *Use default JRE* e *Create separate folders...* e clique em **FINISH**. NÃO confirme a mudança de perspectiva.

4. Crie uma nova classe com método main:
   a) Clique com o botão direito no diretório **SRC** ➔ **NEW** ➔ **CLASS**
   b) Digite *`Principal`* como nome da classe, marque a opção *`public static void main`* e clique no botão **FINISH**.
   c) O Eclipse gerará sua classe com o seguinte código:

```
Principal.java
1
2  public class Principal {
3
4      /**
5       * @param args
6       */
7      public static void main(String[] args) {
8          // TODO Auto-generated method stub
9
10     }
11
12 }
13
```

   d) Observe que, na linha 7, temos o método `main` que é o responsável pela execução de um programa Java.

5. Copie o projeto para a área de trabalho.

6. Troque a Workspace.
   a) Acesse o menu **FILE** ➔ **SWITCH WORKSPACE** ➔ **OTHER** para mudar o local da Workspace.
   b) Digite `C:\Projetos` e Ok.
   c) Reinicie o Eclipse se necessário.

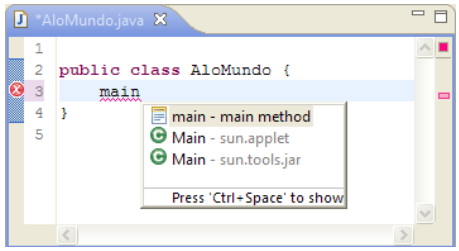
7. Importe o projeto da área de trabalho:
   a) Acesse o menu **FILE** ➔ **IMPORT**.
   b) Escolha **GENERAL** ➔ **EXISTING PROJECTS INTO WORKSPACE** e **NEXT**.
   c) Clique no botão **BROWSE** e escolha a pasta do projeto.
   d) Finalize a importação.
   e) Na `View Navigator` (Project Explorer), exclua a classe *`Principal`*.

8. Crie uma nova classe sem método main:
   a) Crie uma nova classe, digite `AloMundo` como nome da classe, NÃO marque a opção `public static void main` e clique no botão **FINISH**.
   b) A classe gerada pelo Eclipse é a seguinte:

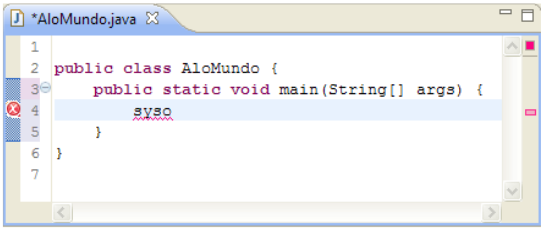
```
AloMundo.java
1
2  public class AloMundo {
3
4  }
5
```

9. Crie o método main através do Content Assist:
   a) Digite `main` e pressione CTRL + ESPAÇO.

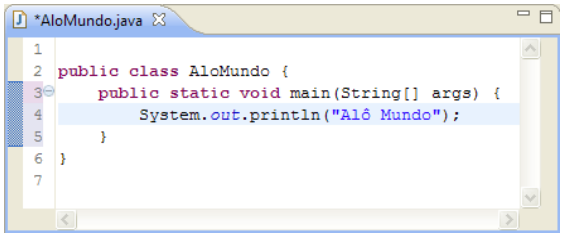
b) Pressione ENTER para gerar o método.



10. Exiba a mensagem "Alô Mundo" na tela:
    a) Digite `Syso`.



    b) Pressione CTRL + ESPAÇO e complete o comando.



11. Altere seu programa para imprimir uma mensagem diferente.

12. Altere seu programa para imprimir duas linhas de texto usando duas linhas de código System.out.

13. Sabendo que os caracteres \n representam uma quebra de linhas, imprima duas linhas de texto usando uma única linha de código `System.out`.

Referência Bibliográfica:

Apostila Caelum - http://www.caelum.com.br/
http://www.globalcode.com.br/home
http://www.k19.com.br/downloads/apostilas-java
http://www.algaworks.com/
Barnes, David J. e Kolling, Michael. Programação Orientada a Objetos com Java. Pearson
Arnold, Ken, Gosling, James e Holmes, David. A Linguagem de Programação Java. Bookman
Harvey M. Deitel, Paul J. Deitel. Java Como Programar. 6ed. Pearson
Flanagan, David. Guia Essencial do Java. Bookman
Martin, Fowler. UML Essencial.
Livro de Informática – Centro Paula Souza